

# Trust and Reputation Management

Omer Rana

School of Computer Science  
and Welsh eScience Centre,  
Cardiff University, UK

# Outline

- 1 Context
  - Defining Trust
- 2 Page Rank
- 3 Ratings
- 4 Reputation Management
  - eBay
  - KaZaA
- 5 Transitive Trust
- 6 Score Managers

# Open Environments

- Move from Client Server to Peer-2-Peer approaches – allowing more autonomy in the network
- Recent emphasis on “Cloud Computing” from `Amazon.com` – EC2 and S3 – and also efforts from Google (Google DFS), Hadoop, IBM “Blue Cloud”
- Enables each participant to be treated as both a client and a server (at different times)
- Deciding who to interact with, or share content from, become important concerns
- There are two key concerns here: (1) secure access, and (2) issue of **trust**

# Defining Trust

- “Trust is the firm belief in the reliability/truth/strength of an entity” – Oxford English Dictionary
- “Trust is a particular level of subjective probability with which an agent assesses that another agent will perform a given action” – Gambetta’s Trust (“Trust: Making and Breaking Cooperative Relations”, Blake Blackwell, 1998)
- Trust may be:
  - Experience based: evaluate how a trustee performs over multiple interactions (consistency check). Must know what is expected vs. what is delivered
  - Information Gathered from others (PageRank, Eigentrust): generally recommendation based on others. Have some ability to reason about reliability of recommendation and recommender
- Reputation may be viewed as an aggregation of trust for a community

# Google Page Rank

- When searching in Google, a given search string can return a number of possible pages as results – how are these ranked?
- Google assigns an importance to each page – referred to as it's **PageRank** – computed by taking into account the link structure of the network
- The model forming the basis of the PageRank algorithm is a random walk through all the pages of the Internet. Let  $p_t(x)$  denote the possibility of being on page  $x$  at time  $t$ . The PageRank of page  $x$  is expressed as  $\lim(p_t(x))$  for  $t \rightarrow \infty$ .
- To make sure the random walk process does not get stuck, pages with no out-links are assigned artificial links or “teleporters” to all other pages

## Page Rank ... 2

- PageRank is a probability distribution used to represent the likelihood that a person randomly clicking on links will arrive at any particular page. PageRank can be calculated for any-size collection of documents



$$PR(A) = (1 - d) + d \left( \frac{PR(T1)}{C(T1)} + \dots + \frac{PR(Tn)}{C(Tn)} \right)$$

where PR(A) is the PageRank of a page A. PR(T1) is the PageRank of a page T1. C(T1) is the number of outgoing links from the page T1. d is a damping factor in the range  $0 < d < 1$ , usually set to 0.85.

- The PageRank of a web page is therefore calculated as a sum of the PageRanks of all pages linking to it (its incoming links), divided by the number of links on each of those pages (its outgoing links).

# Page Rank: Example

- 4 pages: A, B, C and D – PageRank divided, and equals 0.25 for each page.
- Scenario 1:** B, C, and D only link to A, they would each confer 0.25 PageRank to A. Hence,

$$PR(A) = PR(B) + PR(C) + PR(D)$$

- Scenario 2:** (1) B links to C, A; (2) D links to A, B, C. The *value of the link-votes is divided among all the outbound links* on a page. Thus, page B gives a vote worth 0.125 to page A and a vote worth 0.125 to page C. Now,

$$PR(A) = \frac{PR(B)}{2} + PR(C) + \frac{PR(D)}{3}$$

implies

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$$

$B_u$  all pages linking to page  $u$

# Computing Page Rank



$$A \times X[n] + b$$

- where  $A$  is a matrix corresponding to the connectivity of the entire web – one row for each page;
  - row in  $A$  has the value “ $d/\text{outgoing-links}$ ” for each page it has a link. The outgoing links is the number outgoing for that page
  - $B$  is the constant vector  $(1-d)$
  - $X$  is a vector of the page rank values
- Long time to compute – imagine a matrix the size of the Web squared!
  - Use of optimization techniques to calculate “local” page rank – and then aggregate – various matrix manipulation techniques to accelerate calculation
  - Page Rank can be biased – see *Google Bombing* and other approaches based on *Advertising*

# File Sharing

- In a file sharing system – such as a P2P application – it is possible to flood the network with *inauthentic* files
- An Inauthentic File is one that does not work as expected – or is an incorrect copy of a file that exists elsewhere
- We can therefore define Malicious Peers – as those that share inauthentic files for their own benefit (this could also degrade the performance of the network)
- **Assumption:** Hard to determine authenticity of a file in advance. Could make use of MD5 or similar if authenticity of a file can be verified.

# Need for Ratings

- Need algorithms that allow peers to rate each other based on their past interactions
- For instance, in systems like KaZaA – possible to share virus-infected or Trojan files
- These ratings may be used to find **reputable** peers – and this info. may be **propagated**
- In some ways, similar to the rating system on **Amazon.com** and the approach used by popular search engine **Google**

See `epinions.com` – and other work on social networks.

# eBay

- eBay is a system for conducting on-line auctions – and provides a “Feedback System”
- Buyers and sellers rate each other
  - Positive: +1
  - Neutral: 0
  - Negative: -1
- A users reputation is the total number of Positive Remarks received
- This approach is now also adopted by a number of other on-line purchase/selling systems
- However there is no support for “non-repudiation” – i.e. if you provide a service which you agreed, no way to confirm this

# KaZaA

- A popular file sharing application – mainly for music files
- KaZaA uses “Integrity Rating”
  - Individual peers can rate their own files
  - Four levels are supported: Excellent, Average, Poor, Delete File
- Integrity Rated files earn double points towards a peer’s Participation – related to the concept of “free riders”

## KaZaA ... 2

- Participation level reflects how many times a peer has used KaZaA
- Peers who share many Integrity Related files by provided by them (with more download sources) are given greater Reward
- Peers start at a Medium level

$$plevel_i = \frac{uploaded_i}{downloaded_i} \times 100$$

where:  $plevel_i$ : participation level of the  $i^{th}$  peer,  $uploaded_i$ : amount of data (in MBs) that peer  $i$  has uploaded to other peers,  $downloaded_i$ : amount of data (in MBs) that  $i$  has downloaded from other peers

- Non-integrity Rated files are counted as half their size

# KaZaA: key themes

- Designed to reward peers who demonstrate good P2P behaviour – allowing some priority in downloaded at some future time
- Does not punish those who do not or cannot
- No mechanism to allow peers to rate other's files – can only rate their own files
- Can lead to self-benefit – a peer may give every file a high "Integrity Rating" (thus drawing more downloads from others) – even if they are bogus files
- Could lead to an effect where peers are only sharing, highly self-rated bogus files!

# P2P Reputation Systems

- **Self-Policing:** peers themselves enforce shared ethics of the user population – without need for a central authority.
- **Anonymity:** peer reputation should be associated with an opaque identifier (such as the peers Gnutella username) rather than with an externally associated identity (such as a peers IP address).
- **No profit to newcomers:** reputation should be obtained by consistent good behavior through several transactions – non-advantageous to change opaque identifiers.
- **Reduced overhead:** infrastructure overhead for managing system should be reduced – in terms of computation, storage and message overhead.
- **Malicious collectives:** robust to malicious collectives of peers who know one another and attempt to collectively subvert the system.

# Generalising Approach

- Each peer  $i$  rates peer  $j$  – by rating each file download from  $j$
- Rating:  $tr(i, j) = 1$  (positive), or  $tr(i, j) = -1$  (negative) – depending on whether  $i$  was able to download an authentic file from  $j$
- A local trust value  $s_{ij}$  as the sum of the ratings of the individual transactions that peer  $i$  has downloaded from peer  $j$  –  $\sum tr(i, j)$

## Generalising Approach ... 2

- Equivalently, each peer  $i$  can store the number of satisfactory transactions it has had with peer  $j$ . **Local Trust Value** ( $s_{ij}$ ) is the sum of the ratings of all of  $i$ 's interactions with  $j$

$$s_{ij} = \text{sat}(i, j) - \text{unsat}(i, j)$$

- Challenge: how to aggregate  $s_{ij}$  without a centralized storage and management facility – with a wide view of the overall system.
- **Transitive trust**: peer  $i$  will have a high opinion of those peers who have provided it authentic files – and also trust their opinions.

# Aggregating

- To aggregate these values across the network – we normalise them (so that malicious peers do not assign arbitrarily high trust values to other malicious peers)

$$c_{ij} = \frac{\max(s_{ij}, 0)}{\sum_j \max(s_{ij}, 0)}$$

$c_{ij}$  is the **normalised local trust value**

- Normalisation means that all a peer's global trust value will be between 0 and 1
- If  $\sum_j \max(s_{ij}, 0) = 0$  then  $c_{ij}$  is undefined – suggests that either peer  $i$  does not download from anyone else, or assigns a score of 0 to everyone else (i.e. does not trust other peers)
- Useful to define a group of “pre-trusted” peers – these may be the peers that first join the network

# Asking Opinions

- To aggregate normalised local trust values – peer  $i$  can ask its acquaintances about their opinions of other peers
- These opinions are then used to weigh the trust that peer  $i$  places in its own acquaintances

$$t_{ik} = \sum_j c_{ij} c_{jk}$$

$t_{ik}$ : trust that peer  $i$  has in peer  $k$  based on asking each peer  $j$  that it has interacted with

# Random Surfer Model

- Consider an “agent” searching for reputable peers,
- it can crawl the network using the following rule: at each peer  $i$ , it will crawl to peer  $j$  with probability  $c_{ij}$
- After crawling for a while in this manner, the agent is more likely to be at reputable peers than unreputable peers.
- Also the basis for a referral mechanism – and used in approaches such as Google **Page Rank**.

# Practical Considerations

- **Pre-trusted peers:** Some peers in the network that are known to be trustworthy – first few peers to join a network likely to have less motivation to destroy the network they built. Hence, use some distribution  $\vec{p}$  over pre-trusted peers<sup>1</sup>. For example, if some set of peers  $P$  are known to be trusted, we may define  $p_i = 1/|P|$  if  $i \in P$ , and  $p_i = 0$  otherwise.)
- **Inactive peers:** if peer  $i$  does not download from anybody else, or if it assigns a zero score to all other peers, set  $c_{ij} = p_j$ . That is, if peer  $i$  does not know anybody, or does not trust anybody, he will choose to trust the pre-trusted peers.

## Practical Considerations 2

- **Malicious collective:** a group of malicious peers who know each other, who give each other high local trust values and give all other peers low local trust values in an attempt to subvert the system and gain high global trust values. Need to:
  - Detect the existence of such a grouping – by determining message exchanges between peers.
  - Ensure that there are some pre-trusted or non-collective peers in the grouping.

# Dealing with False Reporting

- In EigenTrust, each peer is allowed to compute and report its own trust value to other peers on the network
- Malicious peers can easily report false trust values.
- Two approaches:

trust value of a peer must not be computed by and reside at the peer itself – using another peer to compute trust value  
trust value of one peer in the network will be computed by more than one other peer

- Solution: Use of a distributed hash tables (DHT) to assign **score managers**

# Score Managers

- **Score managers:** peers that calculate the global trust values of other peers. Now,  $M$  peers (score managers of a peer  $i$ ) compute it's trust. A majority vote on the trust value then settles conflicts arising from a number of malicious peers being among the score managers.
- Mapping to score managers – use of a hash function  $h$ , where  $h(key) = value$  – keys can be file names or addresses.
- Distributed Hash Tables (DHTs) deterministically map keys (file names) into points in a logical coordinate space. At any time, the coordinate space is partitioned dynamically among the peers in the system such that every peer covers a region in the “coordinate space.
- To assign a score manager, a peers unique ID (such as IP address/port number) is mapped by a hash function onto a point in the network's co-ordinate space
- The peer which currently is present (or is assigned to cover) this part of the co-ordinate space is assigned as a score manager for that peer

## Score Managers 2

- Knowing the unique ID of a peer, it is possible to locate the score manager to get the global trust for a peer
- Each peer is also a score manager, and each peer has a set of Daughter peers for which it is the score manager
- Each Daughter peer provides its score manager with a set of peers that the Daughter peer has tried to download files from
- Score manager advantages:
  - Anonymity
  - Randomisation
  - Redundancy

# XRep

- Reputations are calculated based on votes
- Each peer maintains a “resource repository” and a “servent repository”
- Peer  $i$  polls its peers
- Peer  $i$  discards bad votes, and clusters/weights the remaining votes
- Based on votes received,  $i$  contacts the most reputable peer that hosts the file it wants
- After checking that file exists on the reputable peer,  $i$  downloads the file

# General Themes

- Still open research area – hard to build an efficient system
- Reputation must decay over time – things done recently more important than those done long time ago
- Use of social networking as a means to enhance reputation in a “community”
- Need ways to exchange reputation scores across different P2P systems